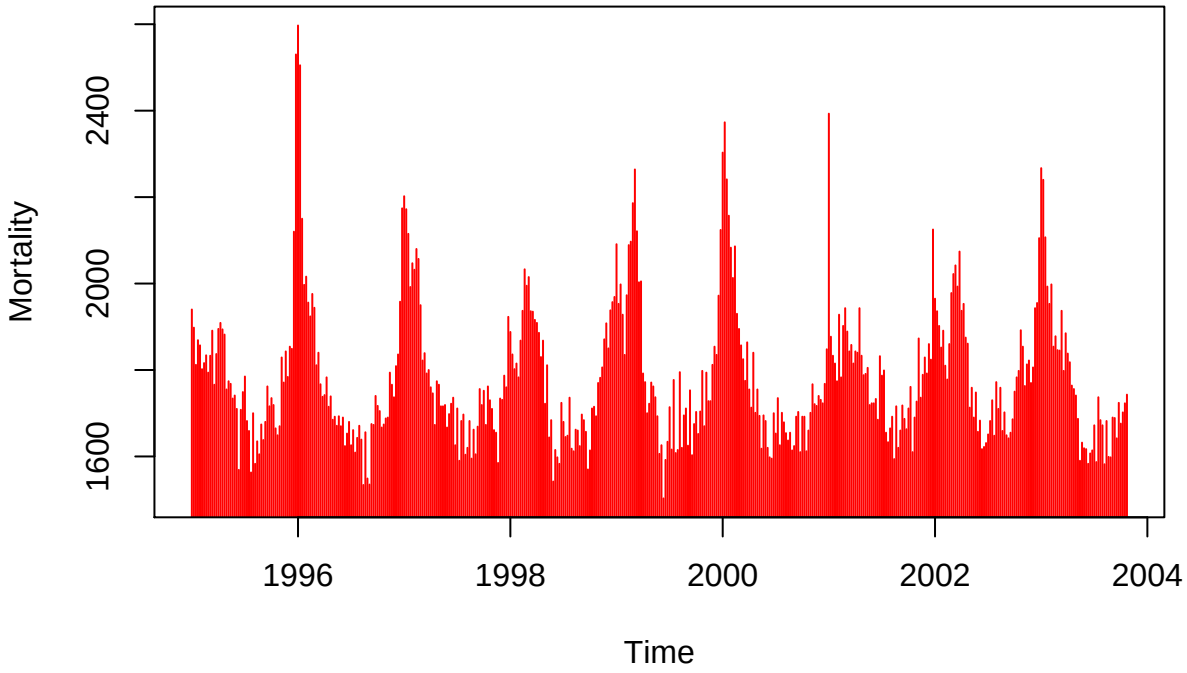
732A99 ComputerLab2 Block2

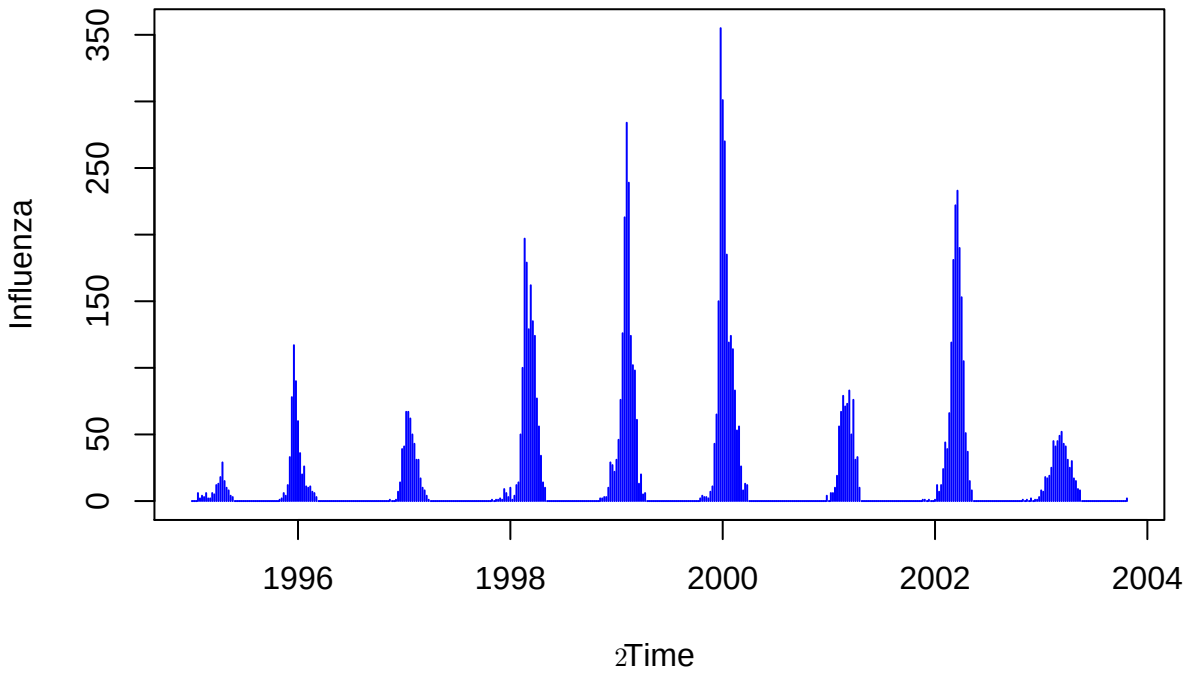
Namita Sharma

12/15/2019

Assignment 1. Using GAM and GLM to examine the mortality rates

1.1 Time series plots





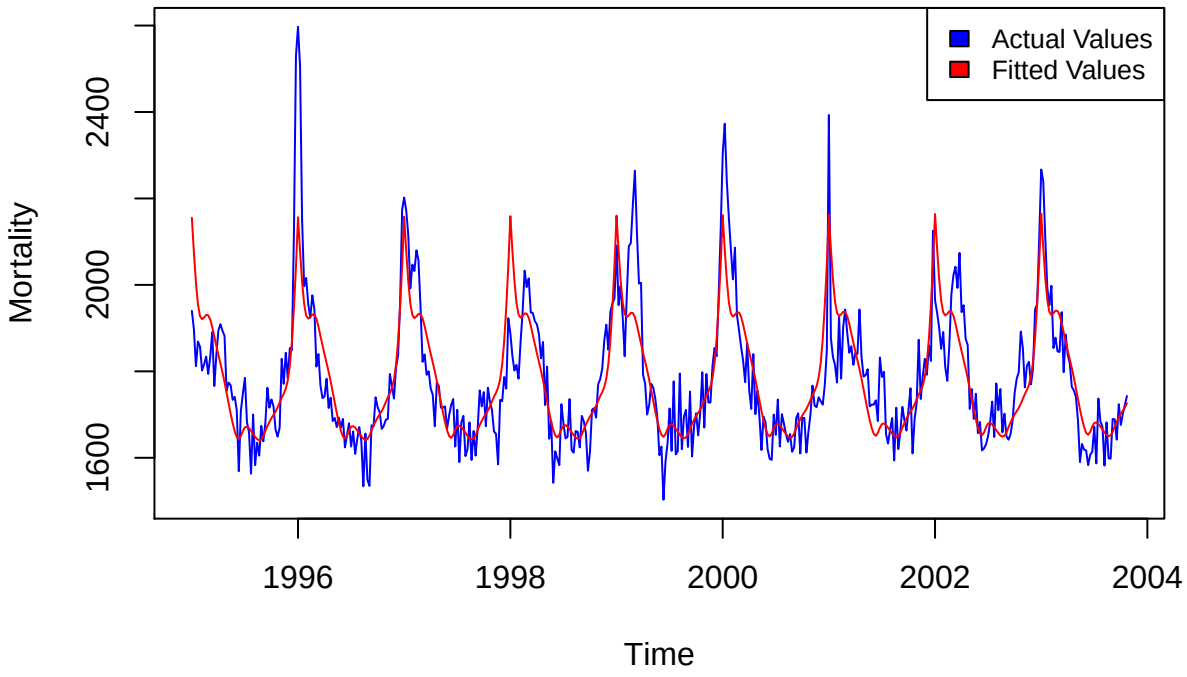
It is seen from the first plot (Mortality vs Time) that the mortality rates peak around the start and end of each year. Similarly, from the second graph (Influenza vs Time) we see the same trend, i.e. the amounts of Influenza cases in any given year is the highest during the beginning and end of that year. This may be attributed to the fact that the cold temperatures in winter are found to be a contributing fator to the spreading of flu in some studies.

Thus an increase in influenza outbreak corresponds to an increase in the mortality rates. However, we cannot comment on the influenza related deaths because the same factors contributing to an increase in the influenza rates could be responsible for the increase in the mortality rates due to other diseases.

1.2 Mortality as a linear function of Year and spline function of Week

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Mortality ~ Year + s(Week, k = length(unique(influenza$Week)))
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -680.598   3367.760  -0.202    0.840
## Year           1.233      1.685   0.732    0.465
##
## Approximate significance of smooth terms:
##           edf Ref.df     F p-value
## s(Week) 14.32  17.87 53.86  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 52/53
## R-sq.(adj) =  0.677   Deviance explained = 68.8%
## GCV = 8708.6  Scale est. = 8398.9    n = 459
```

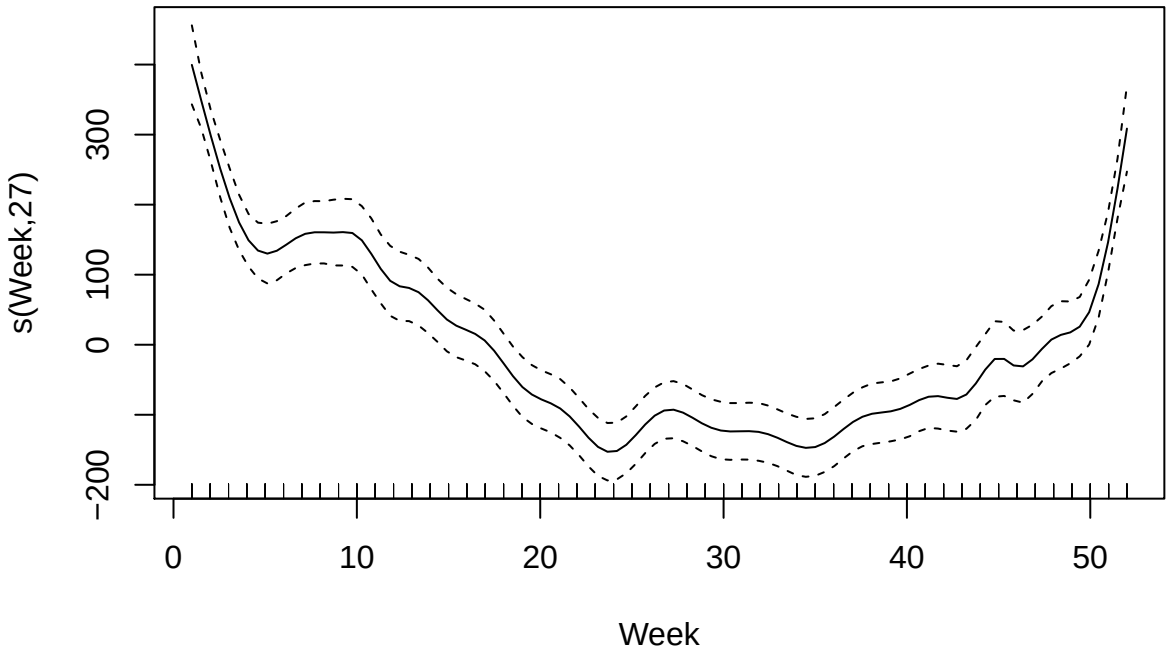
1.3 Plot of predicted and observed mortality against time

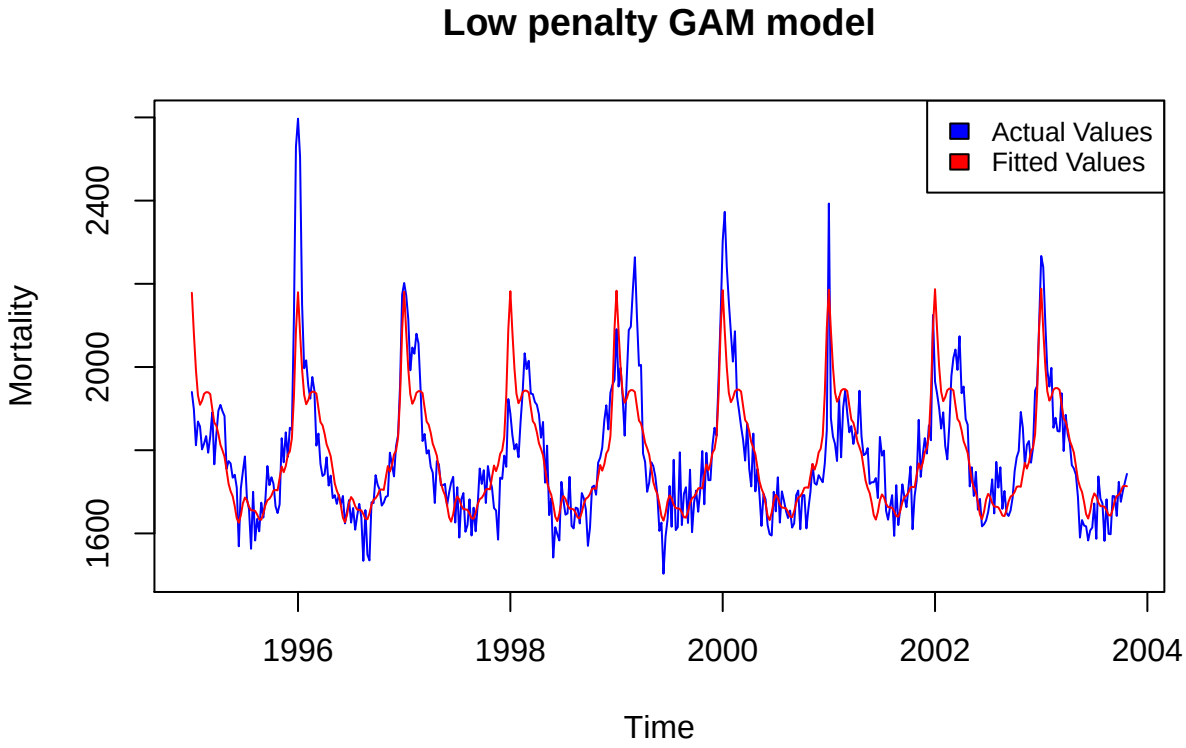


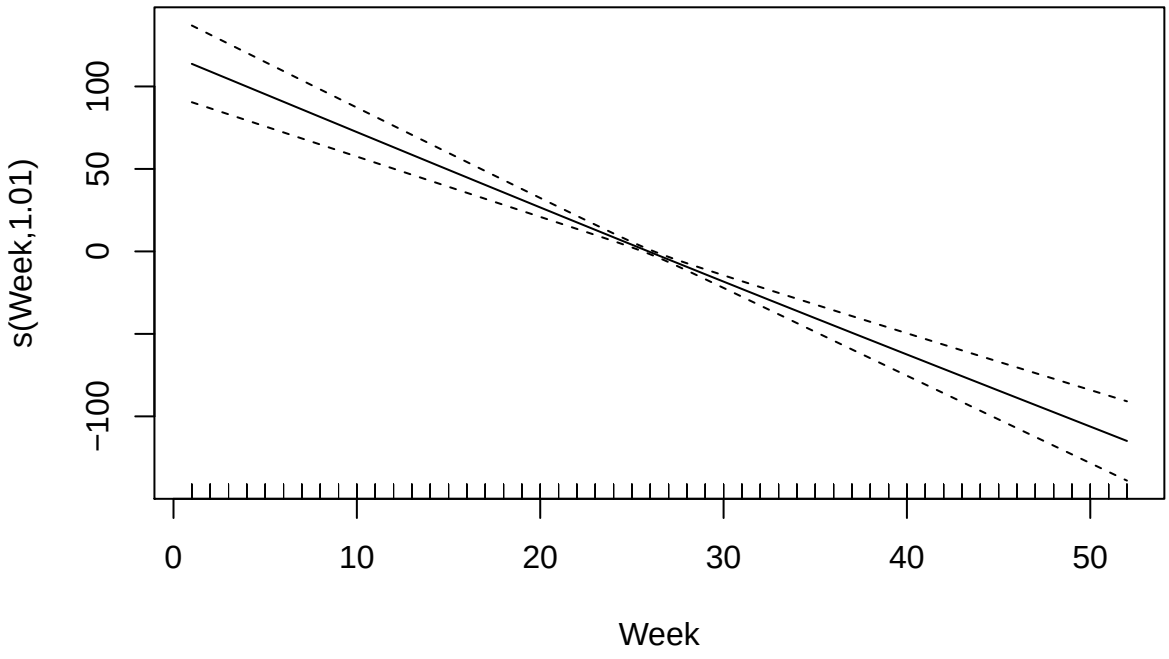
It can be seen form the graph that the mortality predictions are quite good except in the places where mortality rates peak. The predictions when the mortality rates are the highest are not as accurate when the mortality rates are low for some of the years.

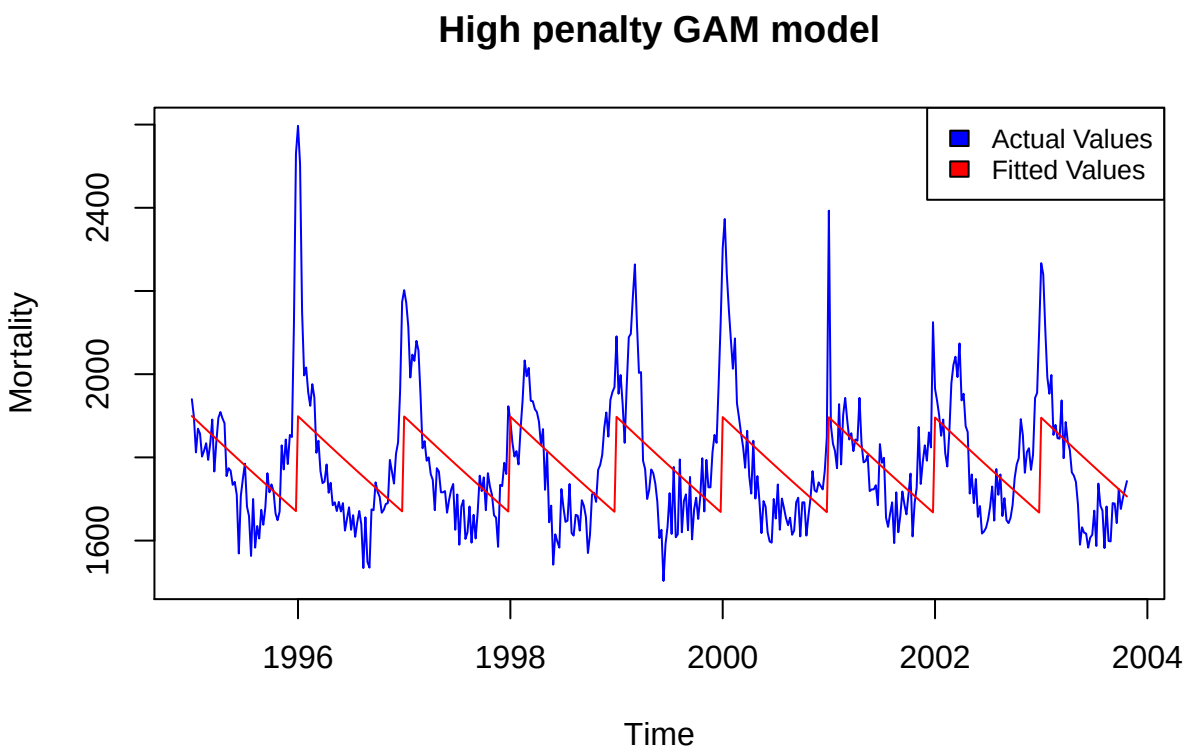
From the GAM model summary, we also see that the spline component (function of week) is the most significant among the linear predictors. We can also see from the plot of the spline component that it models the mortality rates as a function of week quite accurately.

1.4 Effect of penalty factor on estimated deviance and degrees of freedom of GAM model



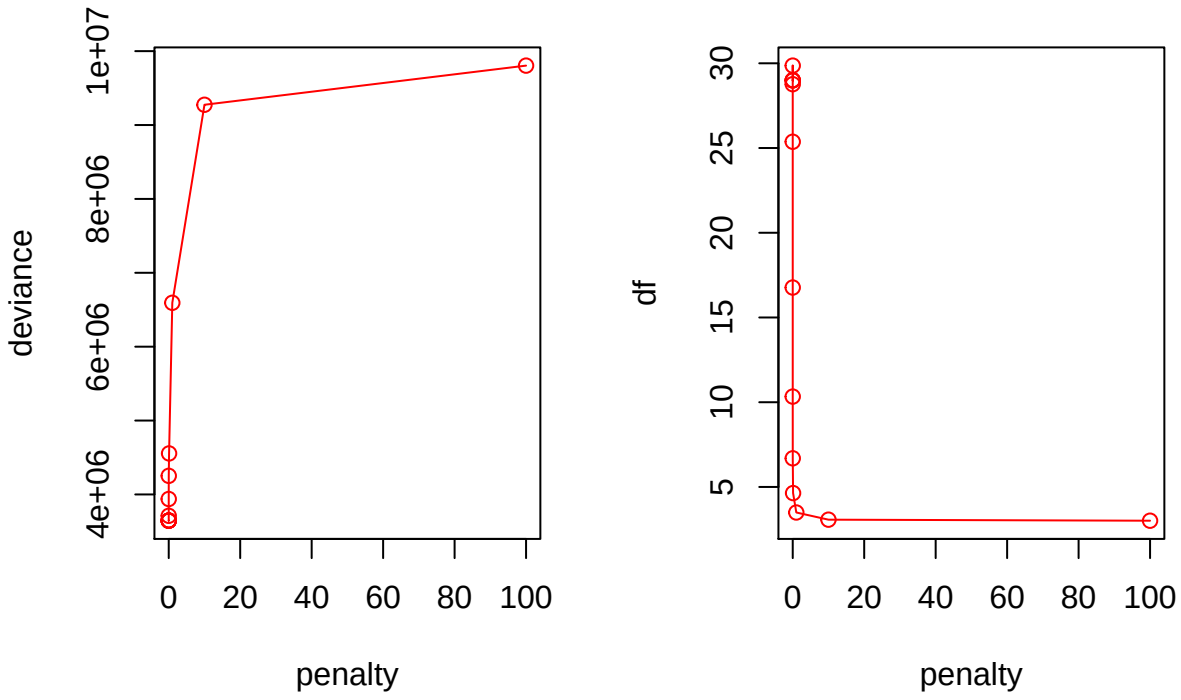



s(Week,1.01)
100
50
0
−100
0
10
20
30
40
50
Week


High penalty GAM model
Actual Values
Fitted Values
Mortality
2400
2000
1600
1996
1998
2000
2002
2004
Time

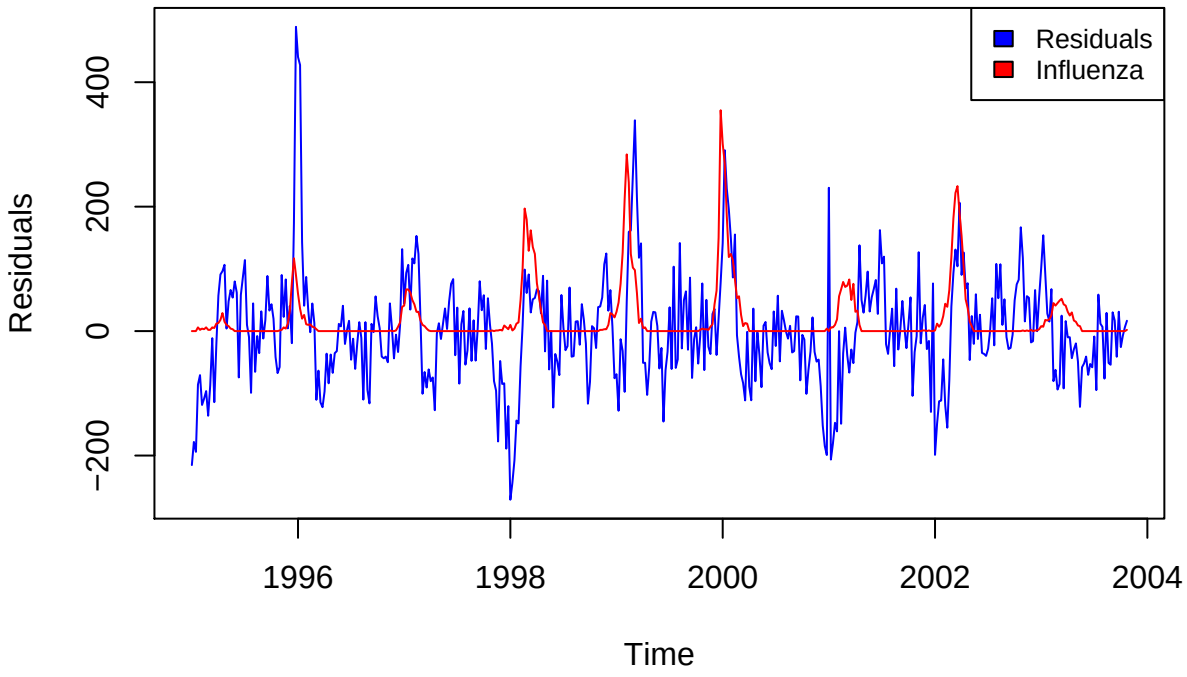
	GCV	Total.df	Deviance
Low Penalty GAM	9049.737	29.000000	3645526
high Penalty GAM	21642.727	3.014497	9804267

We can see that the GAM mode with low penalty is better fitted to our training data. Whereas the GAM model with a very high penalty is not very flexible. The estimated degrees of freedom has reduced to 1 when the model penalized the smooth term heavily to a simple linear relationship. (We can see from the plots of the spline components that increasing the smoothing penalty to a very high value ultimately reduces the spline function to a line) Also, the generalized cross validation score is better and the deviance of the model is lower when the penalty on the smoothing spline is low.



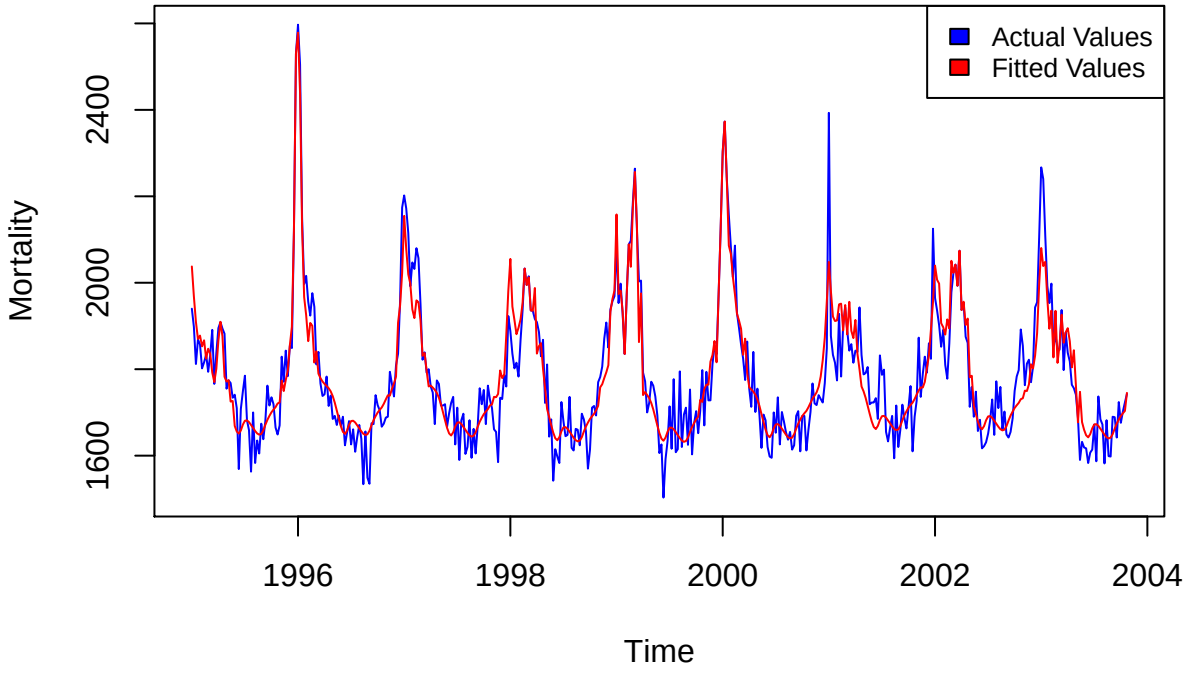
The two plots shown above explain how the penalty factor of the spline function affect the estimated deviance and estimated degrees of freedom of the GAM model in general for different values of penalty. The results confirm that increasing the penalty on the smoothing spline reduces its degrees of freedom making the predictions more linear. Also, increasing the penalty on the spline increases the deviance of the model and hence reduces the goodness of fit.

1.5 Plot of residuals and influenza values against time



The residual values of mortality rate predictions are the high (positive or negative) whenever there are outbreaks in influenza. We do see that the temporal patterns of the residuals are somewhat correlated to the outbreaks of influenza.

1.6 Mortality as an additive function of splines of year, week, and the number of confirmed cases of influenza




```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## Mortality ~ s(Year, k = length(unique(influenza$Year))) + s(Week,
##     k = length(unique(influenza$Week))) + s(Influenza, k = length(unique(influenza$Influenza)))
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1783.765      3.198   557.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                 edf Ref.df      F p-value
## s(Year)       4.587  5.592  1.500   0.178
## s(Week)      14.431 17.990 18.763  <2e-16 ***
## s(Influenza) 70.094 72.998  5.622  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 134/144
```

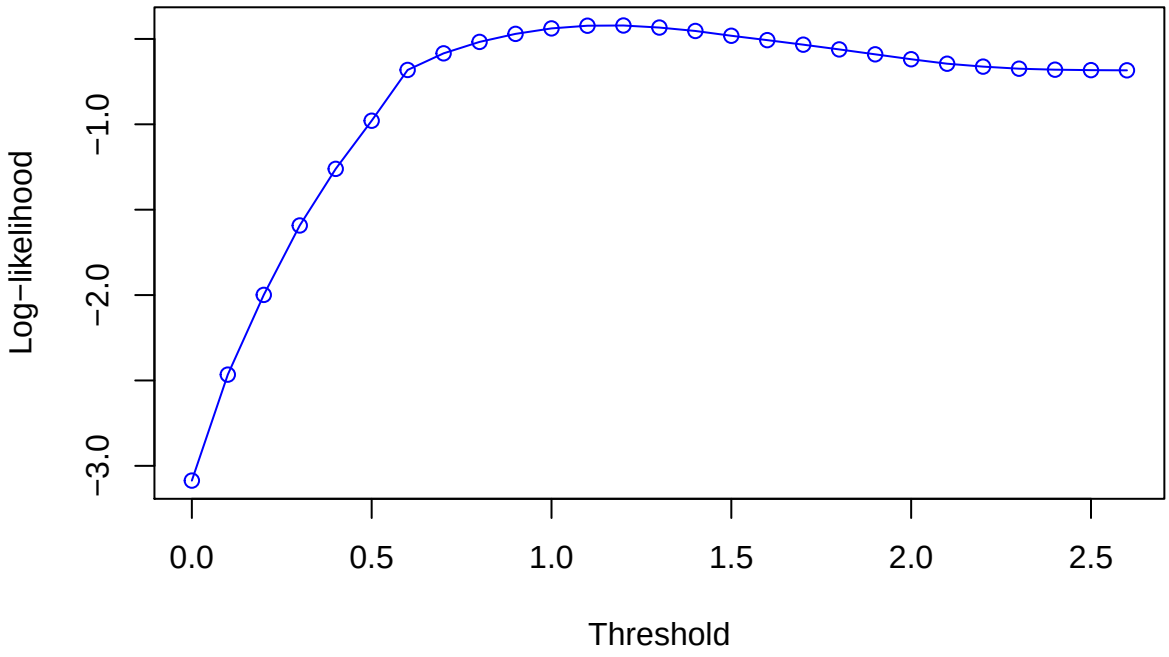
```
## R-sq.(adj) =  0.819   Deviance explained = 85.4%
## GCV = 5840.5  Scale est. = 4693.7    n = 459
```

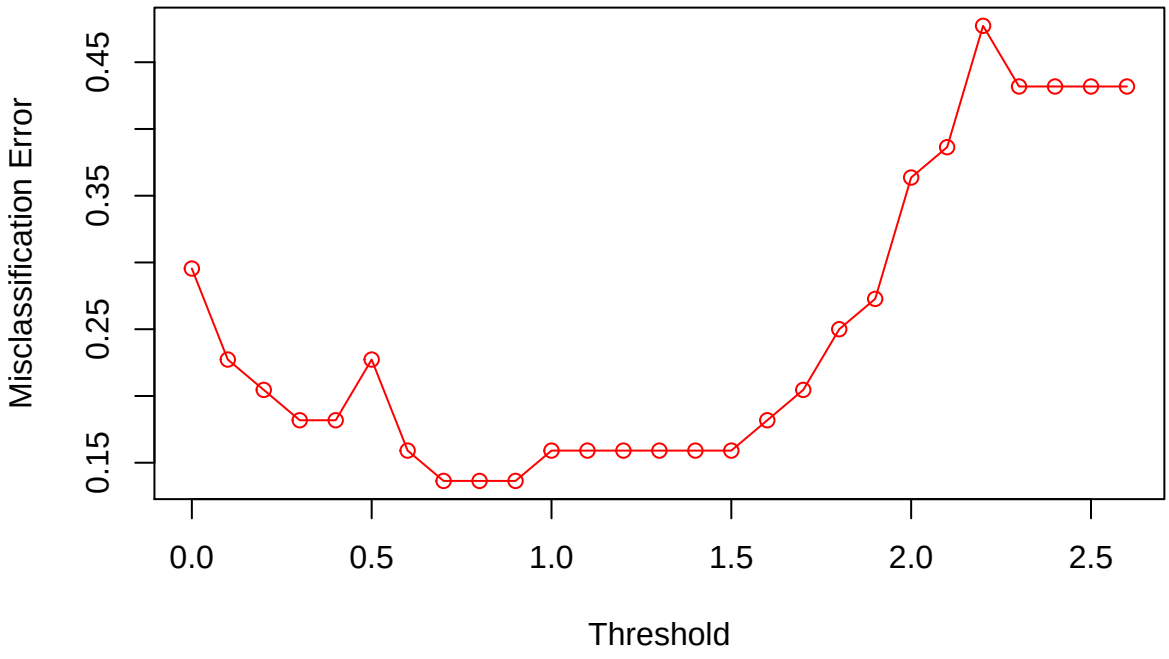
	GCV	Total.df	Deviance
GAM with 3 splines	5840.177	97.58037	1731415
GAM with 1 spline	8708.581	19.86717	3718012

We can seen from the model summary that the spline components of week and influenza cases are both significant terms among other linear predictors. The generalized cross validation score of the GAM model has imcreased and the deviance reduced when compared to the GAM model with only one spline component of week. From the graph of actual and predicted mortality rates, it can be seen that the predictions have improved in the instances where the mortality rates peak. Hence, it can be concluded that this GAM model has the best overall fit.

Assignment 2. High-dimensional methods

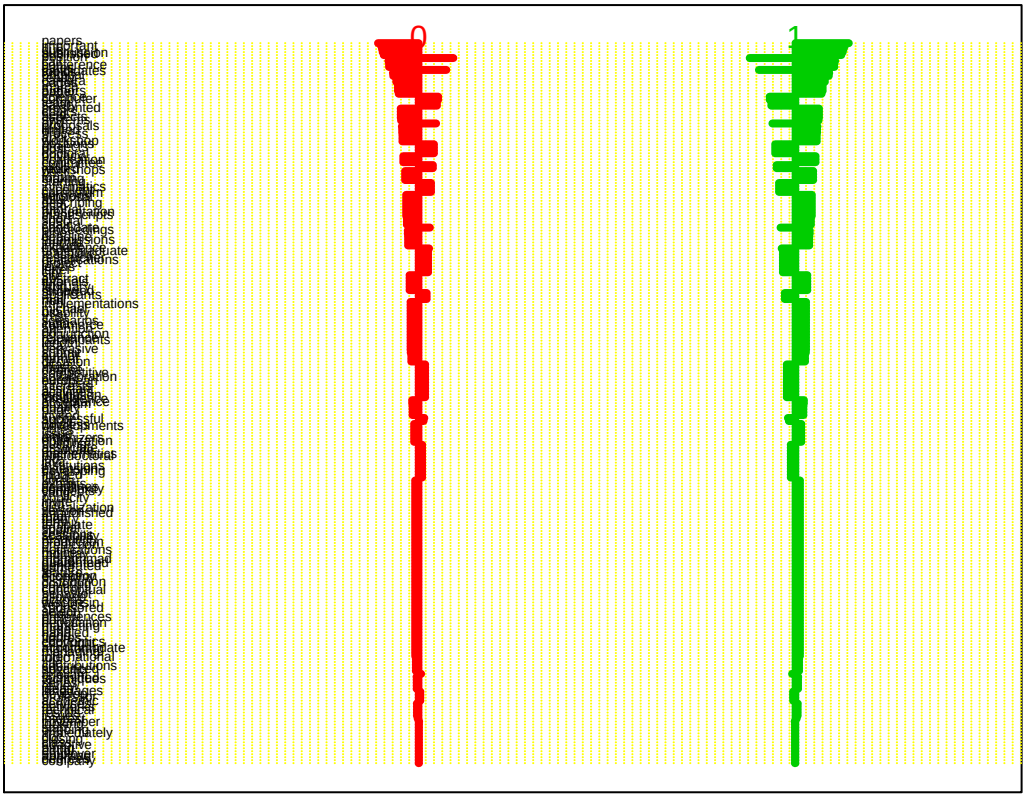
2.1 Nearest Shrunken Centroid Classification





using cross-validation, we see from the misclassification error plot that the error rate reduces to minimum at the threshold values 0.7, 0.8 and 0.9. And among these three thresholds, 0.9 has the highest log-likelihood and fewer non zero features. Hence, we select 0.9 to be the optimum threshold value which gives us 231 model features.

threshold	nonzero	error	loglik
0.7	615	0.14	-0.58
0.8	550	0.14	-0.52
0.9	243	0.14	-0.47



The centroid plot provides the list of all 231 features in the model along with their scores. Listed below are the 10 most contributing features among all 231.

x
papers
important
due
submission
published
position
call
conference
dates
candidates

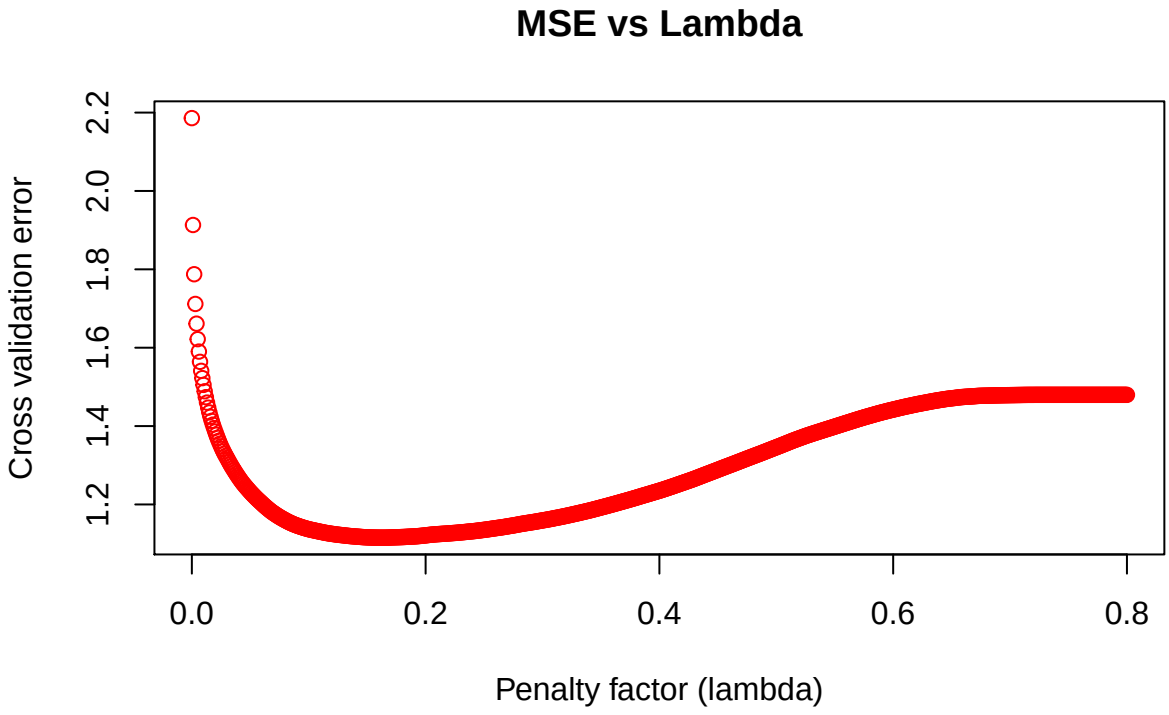
The test error of the model are summarized below. The predictions seem to be quite good.

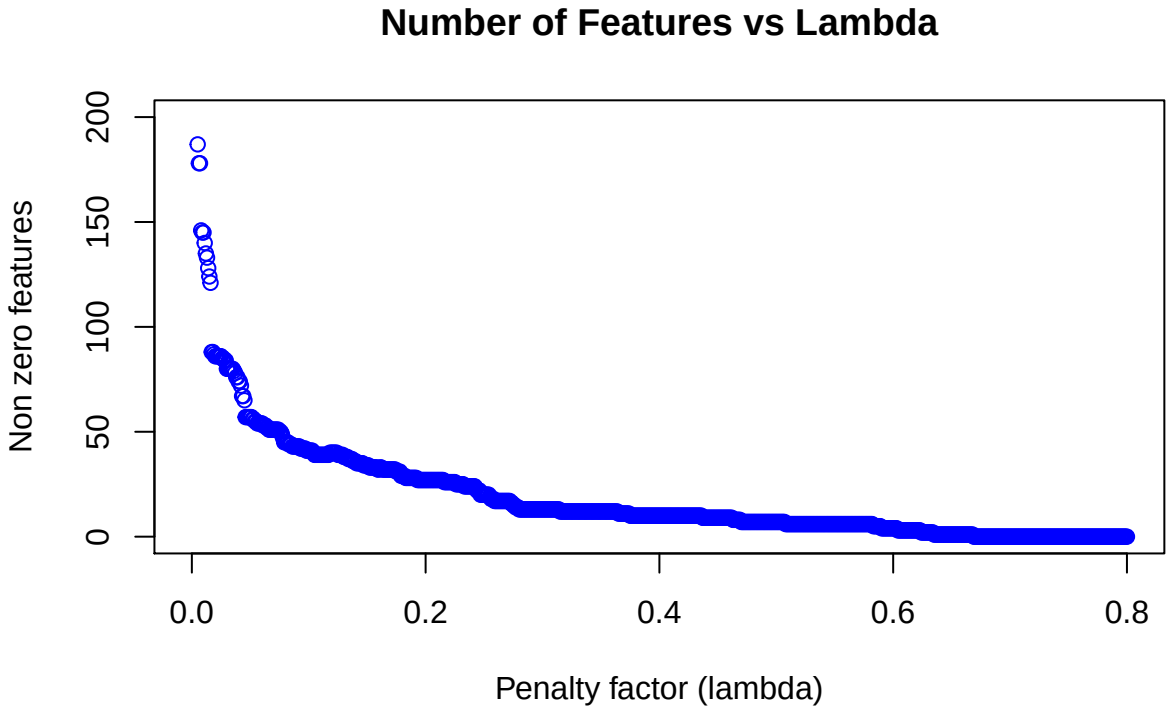
Table 5: Test Confusion Matrix

	0	1
0	10	0
1	1	9

Misclassification.Rate
0.05

2.2 (a) Elastic net regression





	Penalty	Non.zero.Features	Cross.Validation.Error
Lambda.1se	0.455	9	1.294291
Lambda.min	0.162	33	1.115202

We select the optimum λ as 0.162 because it gives us the lowest cross validation error. Hence, the optimum model has 33 features with test error of about 0.1 as shown below. It can be noted that, even though the test error of the elastic net model is slightly greater than the shunken centroid model, the former has far fewer features in the model and is much less complex than the latter.

Table 6: Test Confusion Matrix

	0	1
0	10	0
1	2	8

Misclassification.Rate
0.1

2.2 (b) Support Vector Machine with Vanilladot kernel

Table 7: Test Confusion Matrix

	0	1
0	10	0
1	1	9

Misclassification.Rate
0.05

The test error for the SVM model is 0.05 which is quite less compared to the elastic net model. Also the number of selected features in the model are relatively fewer. Summarized below a tabular comparison of the test error rates and the number of selected features in all of the three models shrunken centroid, elasticnet and SVM.

	Test.Error	Non.zero.Features
Shrunken Centroid	0.05	243
Elastic Net	0.10	33
Support Vector Machine	0.05	43

Based on the above comparison, Elasticnet or SVM model maybe preferred over the shrunken centroid model because of the low error rates and relatively lower model complexity.

2.3 Benjamini-Hochberg method

	pvalues	BH
papers	0.0000000	0.0000005
submission	0.0000000	0.0000019
position	0.0000000	0.0000129
published	0.0000002	0.0002157
important	0.0000003	0.0002860
call	0.0000004	0.0003122
conference	0.0000005	0.0003420
candidates	0.0000009	0.0005062
dates	0.0000014	0.0006576
paper	0.0000014	0.0006576
topics	0.0000051	0.0021665
limited	0.0000079	0.0030986
candidate	0.0000119	0.0043063
authors	0.0000215	0.0063314

	pvalues	BH
camera	0.0000210	0.0063314
ready	0.0000210	0.0063314
phd	0.0000338	0.0091405
projects	0.0000350	0.0091405
org	0.0000374	0.0092605
chairs	0.0000586	0.0137773
due	0.0000649	0.0138683
original	0.0000649	0.0138683
notification	0.0000688	0.0140696
salary	0.0000797	0.0156184
record	0.0000909	0.0164390
skills	0.0000909	0.0164390
held	0.0001529	0.0266303
team	0.0001758	0.0295146
apply	0.0002166	0.0308409
committee	0.0002117	0.0308409
international	0.0002296	0.0308409
pages	0.0002007	0.0308409
proceedings	0.0002117	0.0308409
strong	0.0002246	0.0308409
workshop	0.0002007	0.0308409
degree	0.0003762	0.0453942
excellent	0.0003762	0.0453942
post	0.0003762	0.0453942
presented	0.0003765	0.0453942

Using two-sided t-tests, we see that 281 out of 4702 features have p-values < 0.05 and are therefore significant. On implementing Benjamini-Hochberg method, we see that the p-values are adjusted such that some of the significant features become non-significant. Only the 39 features listed above are selected using the BH method. We see that the top 10 listed features are the same as those of the shrunken centroid model.

Appendix

```
#########################################################################
# Assignment 1. Using GAM and GLM to examine the mortality rates
#########################################################################
library("mgcv")

# Import influenza data
influenza <- xlsx::read.xlsx(
  file      = "C:/Users/namit/Downloads/Machine Learning/Lab2 Block2/Influenza.xlsx",
  sheetName = "Raw data",
  header    = TRUE
)

#-------------------------------------------------------------------------
# 1.1 Time series plots
plot(x    = influenza$Time,
     y    = influenza$Mortality,
```

```r
     xlab = "Time",
     ylab = "Mortality",
     col  = "red",
     type = "h")

plot(x    = influenza$Time,
     y    = influenza$Influenza,
     xlab = "Time",
     ylab = "Influenza",
     col  = "blue",
     type = "h")

#-----------------------------------------------------------------------
# 1.2 Mortality as a linear function of Year and spline function of Week
gam <- gam(Mortality ~ Year + s(Week, k = length(unique(influenza$Week))), data = influenza)
summary(gam)

#-----------------------------------------------------------------------
# 1.3 Plot of predicted and observed mortality against time
plot(x    = influenza$Time,
     y    = influenza$Mortality,
     xlab = "Time",
     ylab = "Mortality",
     col  = "blue",
     type = "l")
points(x = influenza$Time, y = gam$fitted.values, col = "red", type = "l")
legend("topright", c("Actual Values", "Fitted Values"), fill = c("blue", "red"), cex = 0.8)

#-----------------------------------------------------------------------
# 1.4 Effect of penalty factor on estimated deviance and degrees of freedom
# of GAM model
# GAM with low penalty
gam_lowp  <- gam(Mortality ~ Year + s(Week, k = length(unique(influenza$Week)), sp = 1e-100),
                 data = influenza)
plot(gam_lowp)
plot(x    = influenza$Time,
     y    = influenza$Mortality,
     xlab = "Time",
     ylab = "Mortality",
     main = "Low penalty GAM model",
     col  = "blue",
     type = "l")
points(x = influenza$Time, y = gam_lowp$fitted.values, col = "red", type = "l")
legend("topright", c("Actual Values", "Fitted Values"), fill = c("blue", "red"), cex = 0.8)

# GAM with high penalty
gam_highp <- gam(Mortality ~ Year + s(Week, k = length(unique(influenza$Week)), sp = 100),
                 data = influenza)

plot(gam_highp)
plot(x    = influenza$Time,
     y    = influenza$Mortality,
     xlab = "Time",
```

```
     ylab = "Mortality",
     main = "High penalty GAM model",
     col  = "blue",
     type = "l")
points(x = influenza$Time, y = gam_highp$fitted.values, col = "red", type = "l")
legend("topright", c("Actual Values", "Fitted Values"), fill = c("blue", "red"), cex = 0.8)

# Comparison results
results <- data.frame(
  GCV       = c(gam_lowp$gcv.ubre.dev, gam_highp$gcv.ubre.dev),
  Total.df  = c(sum(gam_lowp$edf1), sum(gam_highp$edf1)),
  Deviance  = c(deviance(gam_lowp), deviance(gam_highp)),
  row.names = c("Low Penalty GAM", "high Penalty GAM"))
knitr::kable(results)

# Influence of penalty factor on deviance and estimated degrees
# of freedom
j        <- 1
deviance <- numeric()
df       <- numeric()
penalty  <- cumprod(c(1e-10, rep(10,12)))
for(i in penalty) {
  gam_i <- gam(Mortality ~ Year + s(Week, k = length(unique(influenza$Week)), sp = i),
               data = influenza)
  # Estimated degrees of freedom
  df[j] <- sum(gam_i$edf)
  # Deviance
  deviance[j] <- gam_i$deviance
  j <- j + 1
}

layout(mat = matrix(c(1:2), 1, 2, byrow = TRUE))
# Plot deviance
plot(penalty, deviance, type = "o", col = "red")
# Plot degrees of freedom
plot(penalty, df, type = "o", col = "red")

#------------------------------------------------------------------
# 1.5 Plot of residuals and influenza values against time
plot(x    = influenza$Time,
     y    = gam$residuals,
     xlab = "Time",
     ylab = "Residuals",
     col  = "blue",
     type = "l")
points(x = influenza$Time, influenza$Influenza, col = "red", type = "l")
legend("topright", c("Residuals", "Influenza"), fill = c("blue", "red"), cex = 0.8)

#------------------------------------------------------------------
# 1.6 Mortality as an additive function of splines of year, week, and
# the number of confirmed cases of influenza
gam2 <- gam(Mortality ~ s(Year, k = length(unique(influenza$Year))) +
                        s(Week, k = length(unique(influenza$Week))) +
```

```
                        s(Influenza, k = length(unique(influenza$Influenza))),
                        data = influenza)

plot(x    = influenza$Time,
     y    = influenza$Mortality,
     xlab = "Time",
     ylab = "Mortality",
     col  = "blue",
     type = "l")
points(x = influenza$Time, y = gam2$fitted.values, col = "red", type = "l")
legend("topright", c("Actual Values", "Fitted Values"), fill = c("blue", "red"), cex = 0.8)
summary(gam2)

# Comparison results
results <- data.frame(
  GCV       = c(gam2$gcv.ubre.dev, gam$gcv.ubre.dev),
  Total.df  = c(sum(gam2$edf1), sum(gam$edf1)),
  Deviance  = c(deviance(gam2), deviance(gam)),
  row.names = c("GAM with 3 splines", "GAM with 1 spline"))
knitr::kable(results)

#########################################################################
# Assignment 2. High-dimensional methods
#########################################################################
library("pamr")
library("glmnet")
library("e1071")
library("kernlab")

# Import DB world
DBworld <- read.csv2(
  file   = "C:/Users/namit/Downloads/Machine Learning/Lab2 Block2/data.csv",
  header = TRUE)
DBworld$Conference <- as.factor(DBworld$Conference)

# No of observations in the dataset
n = dim(DBworld)[1]

#------------------------------------------------------------------------
# 2.1 Nearest Shrunken Centroid Classification
# Divide dataset into training and test data
RNGversion('3.5.1')
set.seed(12345)
id    = sample(1:n, floor(n * 0.7))
train = DBworld[id, ]
test  = DBworld[-id, ]

# PAMR model
x            <- scale(train[, -4703])
x[is.nan(x)] <- 0
y            <- train[, 4703]
data  <- list(x = t(x), y = as.factor(y), geneid = as.character(1:ncol(x)), genenames = colnames(x))
model <- pamr.train(data, threshold = seq(0, 5, 0.1))
```

```r
# Cross validation
model_cv <- pamr.cv(model, data)

# Cross-validation analysis
plot(x    = model_cv$threshold,
     y    = model_cv$loglik,
     xlab = "Threshold",
     ylab = "Log-likelihood",
     col  = "blue",
     type = "o")

plot(x    = model_cv$threshold,
     y    = model_cv$error,
     xlab = "Threshold",
     ylab = "Misclassification Error",
     col  = "red",
     type = "o")

cv_summary <- cbind.data.frame(
                threshold = model_cv$threshold[8:10],
                nonzero   = model_cv$size[8:10],
                error     = round(model_cv$error[8:10], 2),
                loglik    = round(model_cv$loglik[8:10], 2))

#---------------------------------------------------------------------
# 2.2 (a) Elastic net regression
elasticnet <- cv.glmnet(x            = as.matrix(train[, -4703]),
                        y            = train$Conference,
                        family       = "binomial",
                        alpha        = 0.5,
                        lambda       = seq(0, 0.8, 0.001),
                        type.measure = "deviance")

# Number of features selected for optimum lambda
nzero_1se <- elasticnet$nzero[elasticnet$lambda == elasticnet$lambda.1se]
cvm_1se   <- elasticnet$cvm[elasticnet$lambda == elasticnet$lambda.1se]

nzero_min <- elasticnet$nzero[elasticnet$lambda == elasticnet$lambda.min]
cvm_min   <- elasticnet$cvm[elasticnet$lambda == elasticnet$lambda.min]

# Predictions for optimum lambda
Yfit     <- predict(object = elasticnet, newx = as.matrix(test[, -4703]), type = "class", s = elasticnet
cf_elnet <- table(actual = test$Conference, predicted = Yfit, dnn = c("Truth", "Prediction"))
mc_elnet <- 1 - (sum(diag(cf_elnet)) / sum(cf_elnet))

Yfit     <- predict(object = elasticnet, newx = as.matrix(test[, -4703]), type = "class", s = elasticnet
cf_elnet <- table(actual = test$Conference, predicted = Yfit, dnn = c("Truth", "Prediction"))
mc_elnet <- 1 - (sum(diag(cf_elnet)) / sum(cf_elnet))

# CV score w.r.t lambda
plot(x    = elasticnet$lambda,
     y    = elasticnet$cvm,
     main = "MSE vs Lambda",
```

```
     xlab = "Penalty factor (lambda)",
     ylab = "Cross validation error",
     col  = "red")

# Number of features w.r.t lambda
plot(x    = elasticnet$lambda,
     y    = elasticnet$nzero,
     main = "Number of Features vs Lambda",
     xlab = "Penalty factor (lambda)",
     ylab = "Non zero features",
     ylim = c(0, 200),
     col  = "blue")

#-----------------------------------------------------------------------
# 2.2 (b) Support Vector Machine with Vanilladot kernel
svm_model <- ksvm(x       = Conference ~ .,
                  data    = train,
                  type    = "C-svc",
                  kernel  = "vanilladot")
# Test Error
Yfit      <- predict(object = svm_model, newdata = test[, -4703])
cf_svm_ts <- table(test$Conference, Yfit, dnn = c("Truth", "Prediction"))
mc_svm_ts <- 1 - (sum(diag(cf_svm_ts)) / sum(cf_svm_ts))
nzero_svm <- length(svm_model@coef[[1]])

#-----------------------------------------------------------------------
# 2.3 Benjamini-Hochberg method

ttests <- lapply(DBworld[, -4703], function(x) {
  t.test(x ~ DBworld[[4703]], data = DBworld, alternative = "two.sided", var.equal = FALSE)
})
pvalues <- sapply(X = ttests, FUN = getElement, name = "p.value")
BH <- p.adjust(p = pvalues, method = "BH")

comparison <- as.data.frame(cbind(pvalues, BH))
comparison <- comparison[which(BH < 0.05), ]
comparison <- comparison[order(comparison$BH), ]

knitr::kable(comparison)
```